SQL Lab with Solution

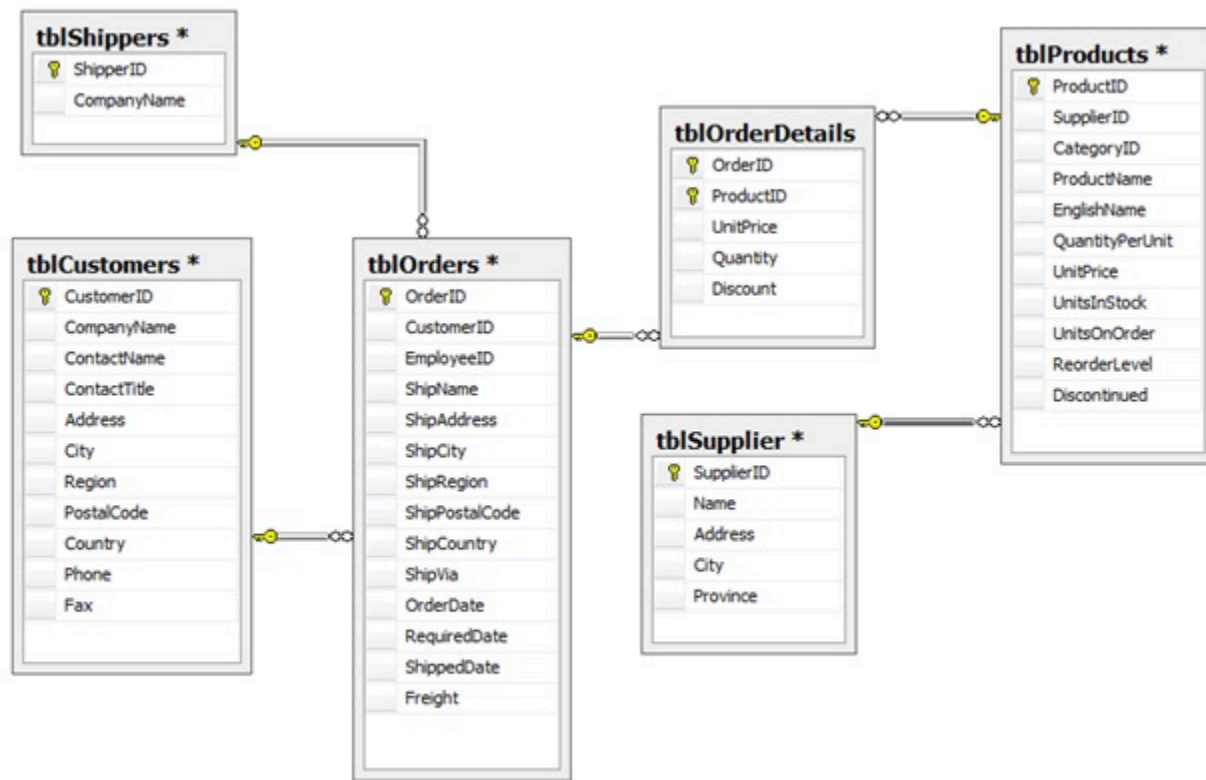Use the following script: Orders.sql.

Part 1 – DDL

Figure C.1. ERD for Orders and Data.

1. Use the script OrdersAndData.sql that creates the tables and adds the data for the Orders and Data ERD in Figure C.1.
2. Create a database called Orders. Modify the script to integrate the PK and referential integrity. Show the CREATE TABLE statements with the modifications including the constraints given in step 3.
3. Add the following constraints:

- tblCustomers table: Country – default to Canada
- tblOrderDetails: Quantity – > 0
- tblShippers: CompanyName must be unique.
- tblOrders: ShippedDate must be greater than order date.

```sql
CREATE DATABASE Orders
Go
Use Orders
Go
```

```sql
Use Orders
Go
CREATE TABLE [dbo].[tblCustomers]
[CustomerID]     nvarchar(5) NOT NULL,
[CompanyName]     nvarchar(40) NOT NULL,
[ContactName]     nvarchar(30) NULL,
[ContactTitle]    nvarchar(30) NULL,
[Address]        nvarchar(60) NULL,
[City]           nvarchar(15) NULL,
[Region]         nvarchar(15) NULL,
[PostalCode]     nvarchar(10) NULL,
[Country]        nvarchar(15) NULL
Constraint    df_country DEFAULT 'Canada',
[Phone]          nvarchar(24) NULL,
[Fax]            nvarchar(24) NULL,
Primary Key (CustomerID)
);
```

```sql
CREATE TABLE [dbo].[tblSupplier] (
[SupplierID]    int NOT NULL,
[Name]          nvarchar(50) NULL,
[Address]       nvarchar(50) NULL,
[City]          nvarchar(50) NULL,
[Province]      nvarchar(50) NULL,
Primary Key (SupplierID)
);
```

```sql
CREATE TABLE [dbo].[tblShippers] (
[ShipperID]     int NOT NULL,
[CompanyName]    nvarchar(40) NOT NULL,
Primary Key (ShipperID),<
```

```
CONSTRAINT uc_CompanyName UNIQUE (CompanyName)
);
```

```
CREATE TABLE [dbo].[tblProducts] (
[ProductID]        int NOT NULL,
[SupplierID]       int NULL,
[CategoryID]       int NULL,
[ProductName]      nvarchar(40) NOT NULL,
[EnglishName]      nvarchar(40) NULL,
[QuantityPerUnit]  nvarchar(20) NULL,
[UnitPrice]        money NULL,
[UnitsInStock]     smallint NULL,
[UnitsOnOrder]     smallint NULL,
[ReorderLevel]     smallint NULL,
[Discontinued]     bit NOT NULL,
Primary Key (ProductID),
Foreign Key (SupplierID) References tblSupplier
);
```

```
CREATE TABLE [dbo].[tblOrders] (
[OrderID]          int NOT NULL,
[CustomerID]       nvarchar(5) NOT NULL,
[EmployeeID]       int NULL,
[ShipName]         nvarchar(40) NULL,
[ShipAddress]      nvarchar(60) NULL,
[ShipCity]         nvarchar(15) NULL,
[ShipRegion]       nvarchar(15) NULL,
[ShipPostalCode]   nvarchar(10) NULL,
[ShipCountry]      nvarchar(15) NULL,
[ShipVia]          int NULL,
[OrderDate]        smalldatetime NULL,
[RequiredDate]     smalldatetime NULL,
[ShippedDate]      smalldatetime NULL,
[Freight]          money NULL
Primary Key (OrderID),
Foreign Key (CustomerID) References tblCustomers,
Foreign Key (ShipVia) References tblShippers,
Constraint valid_ShipDate CHECK (ShippedDate > OrderDate)
);
```

```
CREATE TABLE [dbo].[tblOrderDetails] (
[OrderID]     int NOT NULL,
[ProductID]    int NOT NULL,
[UnitPrice]    money NOT NULL,
[Quantity]     smallint NOT NULL,
[Discount]     real NOT NULL,
Primary Key (OrderID, ProductID),
Foreign Key (OrderID) References tblOrders,
Foreign Key (ProductID) References tblProducts,
Constraint Valid_Qty Check (Quantity > 0)
);
Go
```

Part 2 – Create the Following SQL Statements

1. Show a list of customers and the orders they generated during 2014. Display customer ID, order ID, order date and date ordered.

```
Use Orders
Go
SELECT CompanyName, OrderID, RequiredDate as 'order date', OrderDate as 'date ordered'
FROM tblcustomers  JOIN tblOrders on tblOrders.CustomerID = tblCustomers.CustomerID
WHERE Year(OrderDate) = 2014
```

2. Using the ALTER TABLE statement, add a new field (Active) in the tblcustomer. Default it to True.

```
ALTER TABLE tblCustomers
ADD Active bit DEFAULT ('True')
```

3. Show all orders purchased before September 1, 2012. Display company name, date ordered and total amount of order (include freight).

```
SELECT tblOrders.OrderID, OrderDate as 'Date Ordered', sum(unitprice*quantity*(1-discount))+ freight as
'Total Cost'
```

```
FROM tblOrderDetails join tblOrders on tblOrders.orderID = tblOrderDetails.OrderID
WHERE OrderDate < 'September 1, 2012'
GROUP BY tblOrders.OrderID, freight, OrderDate
```

4. Show all orders that have been shipped via Federal Shipping. Display OrderID, ShipName, ShipAddress and CustomerID.

```
SELECT OrderID, ShipName, ShipAddress, CustomerID
FROM tblOrders join tblShippers on tblOrders.ShipVia = tblShippers.ShipperID
WHERE CompanyName= 'Federal Shipping'
```

5. Show all customers who have not made purchases in 2011.

```
SELECT CompanyName
FROM tblCustomers
WHERE CustomerID not in
(  SELECT CustomerID
FROM  tblOrders
WHERE Year(OrderDate) = 2011
)
```

6. Show all products that have never been ordered.

```
SELECT ProductID from tblProducts
Except
SELECT ProductID from tblOrderDetails
```

OR

```
SELECT Products.ProductID,Products.ProductName
FROM Products LEFT JOIN [Order Details]
ON Products.ProductID = [Order Details].ProductID
WHERE [Order Details].OrderID IS NULL
```

7. Show OrderIDs for customers who reside in London. Use a subquery. Display CustomerID, CustomerName and OrderID.

```
SELECT Customers.CompanyName,Customers.CustomerID,OrderID
FROM Orders
LEFT JOIN Customers ON Orders.CustomerID = Customers.CustomerID
WHERE Customers.CompanyName IN
(SELECT CompanyName
FROM Customers
WHERE City = 'London')
```

8. Show products supplied by Supplier A and Supplier B. Display product name and supplier name.

```
SELECT ProductName, Name
FROM tblProducts JOIN tblSupplier on tblProducts.SupplierID = tblSupplier.SupplierID
WHERE Name Like 'Supplier A' or Name Like 'Supplier B'
```

9. Show all products that come in boxes. Display product name and QuantityPerUnit.

```
SELECT EnglishName, ProductName,  QuantityPerUnit
FROM tblProducts
WHERE QuantityPerUnit like '%box%'
ORDER BY EnglishName
```

Part 3 – Insert, Update, Delete, Indexes

1. Create an Employee table. The primary key should be EmployeeID (autonumber). Add the following fields: LastName, FirstName, Address, City, Province, Postalcode, Phone, Salary. Show the CREATE TABLE statement and the INSERT statements for the five employees. Join the employee table to the tblOrders. Show the script for creating the table, setting constraints and adding employees.

```
Use Orders
CREATE TABLE [dbo].[tblEmployee](
EmployeeID Int IDENTITY NOT NULL ,
FirstName varchar (20) NOT NULL,
```

```
        LastName varchar (20) NOT NULL,
        Address varchar (50),
        City varchar(20), Province varchar (50),
        PostalCode char(6),
        Phone char (10),
        Salary Money NOT NULL,
        Primary Key (EmployeeID)
```

```
        Go
        INSERT into tblEmployees
        Values ('Jim', 'Smith', '123 Fake', 'Terrace', 'BC', 'V8G5J6', '2506155989', '20.12'),
        ('Jimmy', 'Smithy', '124 Fake', 'Terrace', 'BC', 'V8G5J7', '2506155984', '21.12'),
        ('John', 'Smore', '13 Fake', 'Terrace', 'BC', 'V4G5J6', '2506115989', '19.12'),
        ('Jay', 'Sith', '12 Fake', 'Terrace', 'BC', 'V8G4J6', '2506155939', '25.12'),
        ('Jig', 'Mith', '23 Fake', 'Terrace', 'BC', 'V8G5J5', '2506455989', '18.12');
        Go
```

2. Add a field to tblOrders called TotalSales. Show DDL – ALTER TABLE statement.

```
        ALTER TABLE tblOrders
        ADD Foreign Key (EmployeeID) references tblEmployees (EmployeeID)
```

3. Using the UPDATE statement, add the total sale for each order based on the order details table.

```
        UPDATE tblOrders
        Set TotalSales = (select sum(unitprice*quantity*(1-discount))
        FROM tblOrderDetails
        WHERE tblOrderDetails.OrderID= tblOrders.OrderID
        GROUP BY OrderID
```